# Origin Software Environment

*Ruud van der Pas*

*SiliconGraphics*

*European High Performance Computing Team*





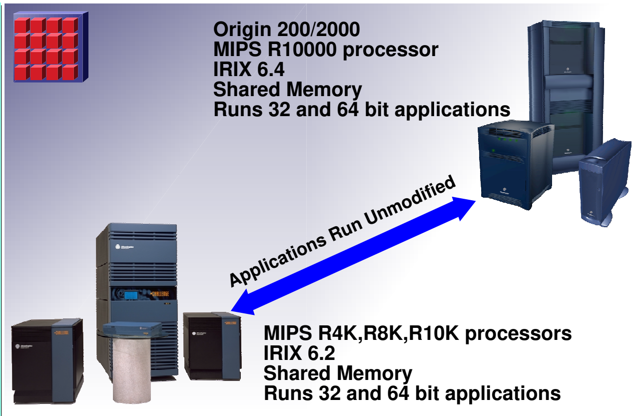
## Topics

- *The IRIX Operating System*
- *IRIX Memory Management Internals*
- *Compilers*
- *Program Development Tools*





## IRIX 6.4: Applications Now!







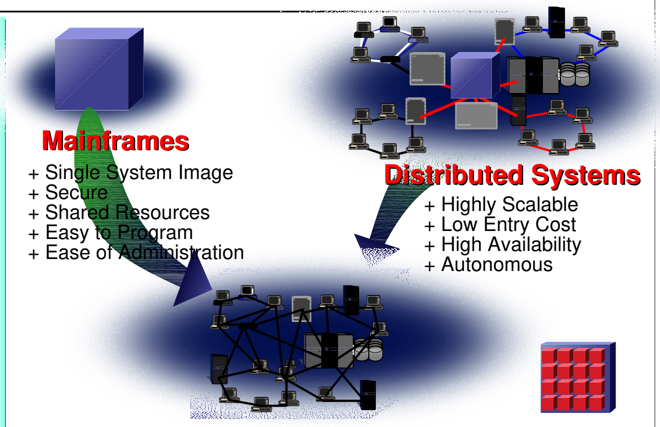

## IRIX 6.4

- ***Evolutionary*** **innovation building on IRIX 6.2**

  **Binary compatible with IRIX 5.3 & 6.2**
- ***Full 64–bit Operating System***

  **With complete support for 32 bit as well**
- ***Integral Checkpoint/Restart***
- ***Automatic*** **memory management localizes data for optimal performance**
- ***Variable Page Sizes***

# Cellular IRIX: Best of both worlds



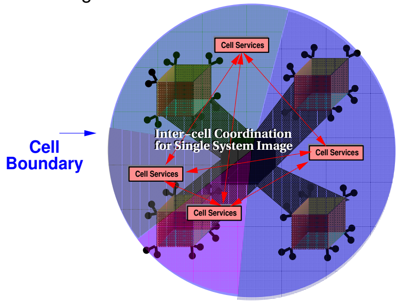





# Cellular IRIX Architecture

**Innovative Multi–Level Cellular OS Architecture**

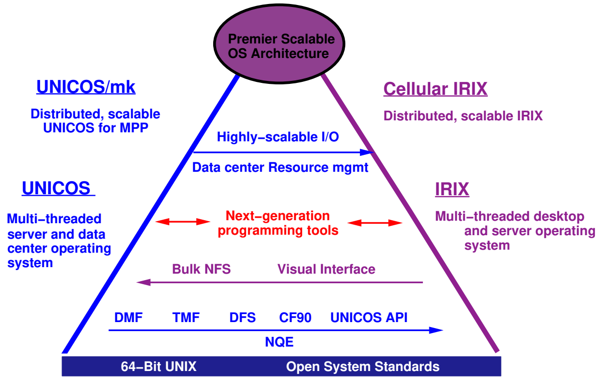
- Full Set of SMP Services Provided Within Each Cell
- Distributed Structure Maintains Single System Image & IRIX Compatibility Across Cells
- Software Configurable Cell Size At Reboot








# Scalable OS *Evolution*








# IRIX 6.4 – Memory Management

- ***Origin has a multi–dimensional topology and notion of memory distance***
- ***IRIX has been modified to exploit this:***
  - *The scheduler is topology aware*
  - *Placement of data is optimized*
- ***We will now illustrate this with an example***

# Don't Worry

**You Do**

**Not**

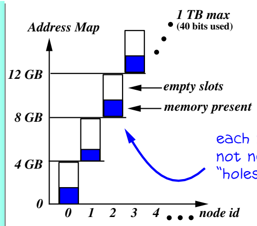
**Need To Use The Following**

(but it is kind of nice to know)





# Memory Management Fundamentals




## *Physical Address Space*

**DIMMs size limits the max installable memory size 256 GB** (4GB/node, 64 nodes for 128 cpus)

each node contains min of 64 MB, but nodes do not need to be numbered sequentially, giving "holes" in the address space.

**To a processor, main memory appears as a *single address space* containing many individually addressable *pages*. Each node has a static portion of this address space. The physical address contains node address in the high byte.**

| 39 – 32 | 31 – 0 |
|---|---|
| Node index 8 bits | Node offset 32 bits (max 4 GB) |

**Node id is assigned to individual nodes dynamically at boot time.**





# It is Just Shared Memory

***At the 80–90% level, Origin is Shared Memory, period***

- **IRIX: Production oriented scalable shared memory OS:**
  - Timesharing, multi–user environment and memory management
  - Transparent device availability
  - Same compilers, tools and programming model
- **Physically:**
  - Memory is distributed and can contain "holes"
  - The scalable interconnect replaces the bus
- **The OS works hard, and most often you won't have to**

**We will now review how memory is handled "under the hood"**

This is mostly "cultural", but the names and concepts creep into the tools that you will actually use





# Memory Management Control (MMC)

**Multi–layered approach using the following mechanisms:**

- *Topology–aware initial placement*
- *Dynamic memory migration*
- *Replication of read–only memory*
- *Schedules processes so that there is a lot of nearby memory*
- *Process scheduler knows memory affinity of processes*
- *Application writers may give placement hints*
- *Selectable and customizable policy choices*

# Memory Management Control (MMC)

## Memory allocation

- Smallest unit of memory allocation is the page
- The OS has default automatic policies
- It is good to know about the automatic OS locality strategies and mechanisms; you don't use them directly though

## Memory locality: a new, finite resource

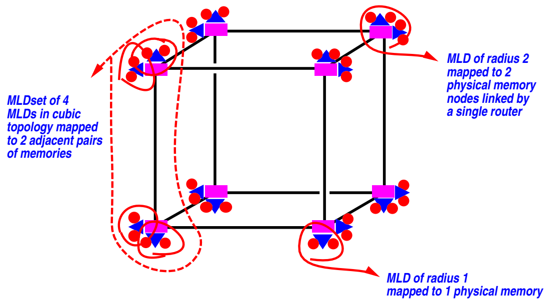
- Users need a measure of control without making it possible for them to choke the system
- This resource needs to be shared fairly

MLDs are an OS virtualization of memory locality





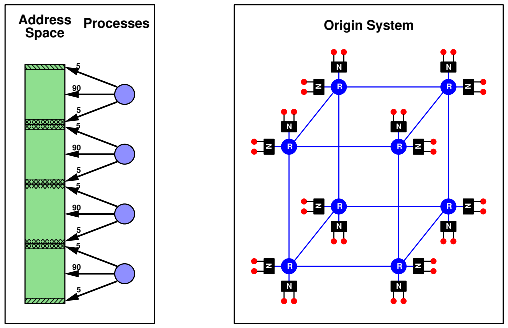
# Memory Locality Domains (MLDs)








# Simple Memory Access Pattern

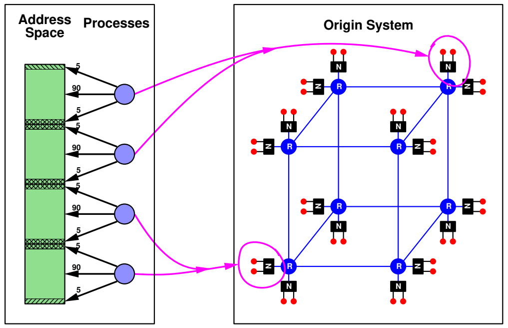



Assume:
90% memory is unshared, 5% shared with another process and 5% shared with a third process

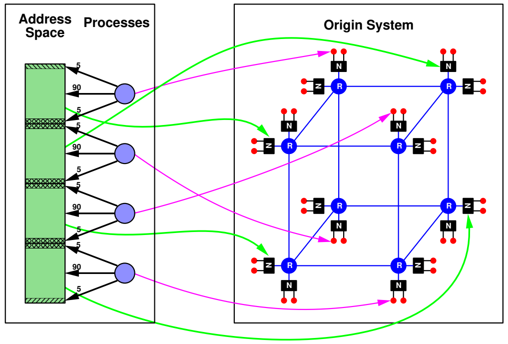




# Non Optimal Placement



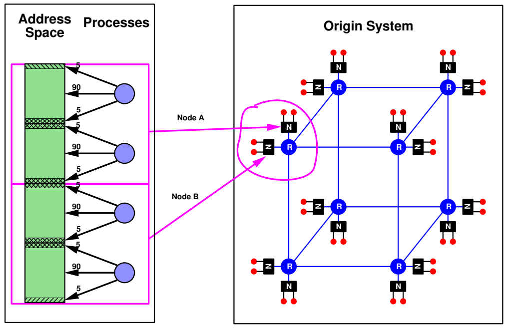
# Random Placement *(worse!)*







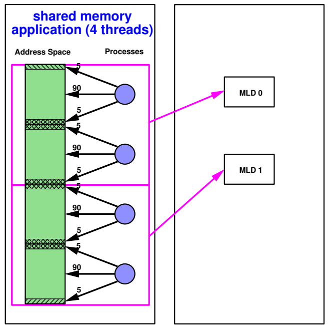

# Ideal Placement







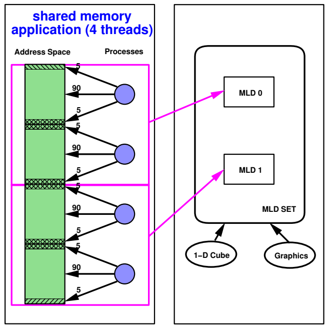

# Memory Locality Domains (MLDs)



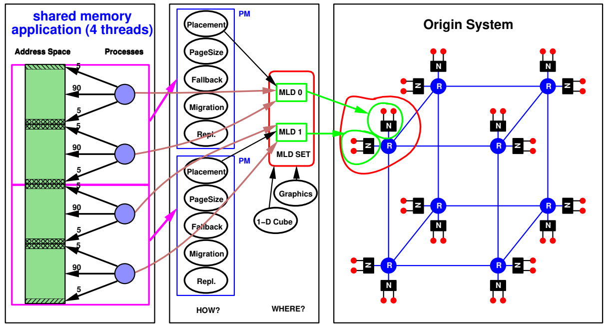





# MLDsets:Topology&Resource Affinity



# Linking Execution Threads to MLDs








# Memory Locality Domains (MLDs)

**MLDs are just abstract memories!**

- **MLDs are sources of memory** with a given radius (in hops)
- **MLDs implement the idea of locality,** while leaving freedom for fair location management to OS
- **created by `mld_create`, returns a `pmo_handle_t`**
- **organized into sets by `mldset_create`**





# Policy Modules

**Policy Modules (PMs) are the way the *OS remembers* what policies have been specified**

- are associated with **address ranges**
- determine **managment policies** for that range with respect to
  *initial placement*
  *pagesize*
  *fallback (when placement and size conflict)*
  *migration*
  *replication*
  *paging*
- **created by `pm_create`** ( returns `pmo_handle_t` )
- **attached** to address range by **`pm_attach`**
- A (settable) **default policy** is pre–attached to every address range, and remains in force unless overridden by explicit `pm_attach`
- **inherited** across **`fork/sproc`, reset** to default on **`exec`**





# MLDset Placement

**Placement = mapping onto the physical machine**

**MLDsets** are placed with **`mldset_place`** requiring:

- **topology**
  TOPOLOGY_FREE: cluster determined by OS
  TOPOLOGY_CUBE: cube or cube subset
  TOPOLOGY_CUBE_FIXED: perfect cube
  TOPOLOGY_PHYSNODES: physical nodes
- **resource affinity**
- **request type** (advisory or mandatory) and returning success or failure

# You can open

your eyes again

# It is Over





## Variable page sizes

- IRIX supports the following page sizes on Origin:

  16K, 64K, 256K, 1M, 4M, 16M

- Mixable; Each address *range* can have an associated page size, defined in the region's policy module (PM)
- They are set through `dplace` or `pm_setpagesize` to change it for a given PM
- In general the *new dplace command* gives you easy access to the OS internals just described
- The kernel attempts to coalesce free memory to have a certain (tunable) fraction at a given size





## MIPSpro Compilers

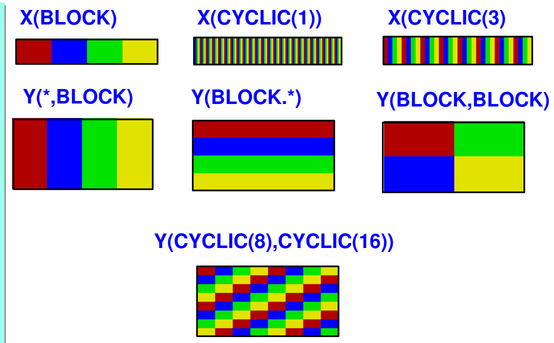
New Additional Compiler Features for Origin:

- Data Distribution Directives to Support:
  - Static distributions
  - Dynamic re-distribution
  - Virtual addresses unchanged
- Affinity scheduling
  - Places thread execution near thread-local data
- Fine-grained data placement control





## Data Distributions – Examples



# Origin Programming Models

- **Shared Memory**
  - Auto–Parallelizing Compilers for F77, F90, C, C++
  - Directives for:
    - Single CPU optimizations (like prefetch)
    - Parallelization control (+PCF directives)
    - Data placement and distribution
- **Message Passing**
  - Tuned versions of MPI and PVM available
  - XMPI for debugging and profiling

*The programming model has remained the same*





# Program Development Tools

- *Debugger (DeveloperMagic)*
- *Profilers (DeveloperMagic, SpeedShop)*
- *Event counters on R10K (perfex, SpeedShop)*
- *Data placement control tool (dplace)*
- *Data movement profiler (dprof, dlook)*
- *Performance Co–Pilot for system monitoring*





# Performance Co–Pilot